Matplotlib una biblioteca Python para generar gráficos interesantes

Yuri Matheus
05/11/2020

Cuando presentamos resultados, además de los números, podemos usar otros elementos para mejor informar. Una buena manera de presentar datos es usar la biblioteca **matplotlib**.

Cuando estamos analizando datos, es común ver y manipular muchos números. Números de ventas, de accesos, de tasa de retorno entre otros tantos.Podemos analizar estos datos de muchas maneras, podemos usar una planilla electrónica, como **Excel**, o **Google Sheets**, podemos utilizar lenguajes como **Python** y **R** para leer nuestros datos y almacenarlos en **estructuras de datos como listas, vectores, data frames**, entre otros.

Estaba utilizando Python para analizar los datos de venta de una tienda de productos electrónicos. En un momento, estaba analizando los datos de venta en el semestre me encontré con la siguiente planilla:

```
Ene - 105235
Feb - 107697
Mar - 110256
Abr - 109236
May - 107589
Jun - 106986
```

Sabemos que esta tabla muestra los datos de venta del semestre, pero ¿qué indican estos datos? ¿Tuvimos un aumento en las ventas? ¿Están bajando?

No tenemos una idea clara de lo que significan estos datos, no podemos interpretarlos fácilmente. ¿Qué podemos hacer para interpretar mejor estos datos?

Una imagen vale más que mil palabras

Ver los datos en la tabla puede no ser la mejor manera de verlos. Tenemos que seguir analizando línea por línea, valor por valor, lo que puede suponer bastante trabajo.

Por supuesto, los números son importantes para que hagamos un análisis preciso, sin embargo, cuando estamos presentando los resultados, además de los números, podemos utilizar otros elementos gráficos para transmitir mejor la información.

Una buena forma de presentar datos es mediante gráficos. Con los gráficos podemos tener una idea general de lo que nos dicen los datos.

Por ejemplo, nuestra planilla de ventas por semestre, podemos crear un gráfico que muestre las **progresiones** de vendas cada mes.

Pero, ¿cómo podemos crear un gráfico?

Conociendo la biblioteca

En Python, hay una biblioteca muy famosa para crear gráficos, la **Matplotlib**. Con ella, podemos elaborar diferentes tipos de gráficos. Para empezar a usar la biblioteca, tenemos que instalarla. Por lo tanto:

```
pip install matplotlib
```

¡Genial! Ya hemos instalado la biblioteca, comencemos a usarla.

Nuestra biblioteca es la `matplotlib`. Queremos usarla para crear, es decir, trazar gráficos. Entonces, le dijimos a Python que importara (`import`) desde la biblioteca `matplotlib` la parte de trazado de gráficos con Python (`pyplot`).

```
>>> import matplotlib.pyplot
```

Creemos un gráfico con nuestra planilla de ventas semestral. Para eso, debemos decirle a **pyplot** cuáles son los meses y cuáles son los valores de cada mes. Una manera de hacerlo es crear dos listas, una con los meses y otra con los valores:

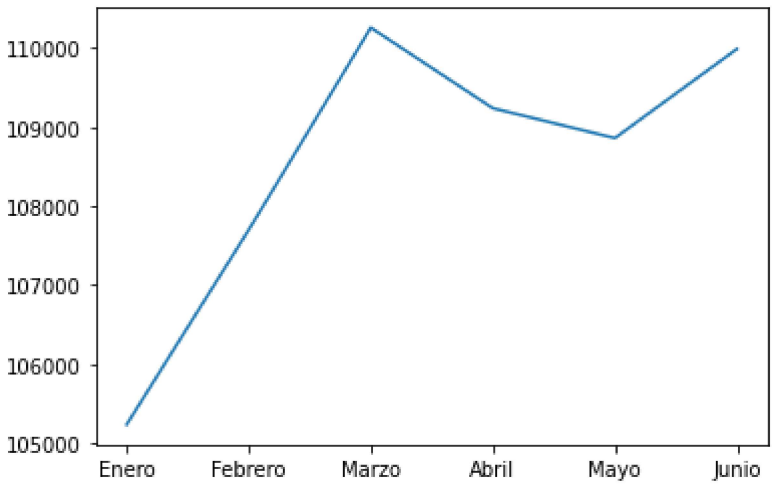
```
>>> meses = ['Enero', 'Febrero', 'Marzo', 'Abril', 'Mayo', 'Junio']
>>> valores = [105235, 107697, 110256, 109236, 108859, 109986]
```

Ahora, basta decirle al `pyplot` que trace (`plot`) nuestro gráfico:

```
>>> matplotlib.pyplot.plot(meses, valores)
```

Le dijimos al `pyplot` que creara nuestro gráfico, pero, ¿dónde está? Nuestro gráfico no apareció. Cuando le decimos al `pyplot` que cree el gráfico, lo construye y lo almacena en una región de la memoria. Para poder ver el gráfico, debemos decirle al `pyplot` que lo muestre (`show()`):

```
>>> matplotlib.pyplot.show()
```



El método **`plot`** crea un gráfico de líneas. Podemos ver que los meses quedaron en la parte horizontal del gráfico, es decir, en el eje X. Mientras que los valores quedaron en la parte vertical del gráfico, es decir, en el eje Y.

Con el gráfico, podemos ver más claramente cómo fueron las ventas en cada mes durante el primer semestre. Está claro que hasta el mes de marzo tuvimos un aumento en las ventas, luego tuvimos una pequeña reducción hasta el mes de mayo y luego volvimos a tener un aumento. ¿Una gran baja?

Si miramos el eje Y del gráfico, veremos que los valores van de **105.000** a **110.000**. Es decir, tenemos cinco mil dolares de diferencia entre la base y el tope del eje Y.

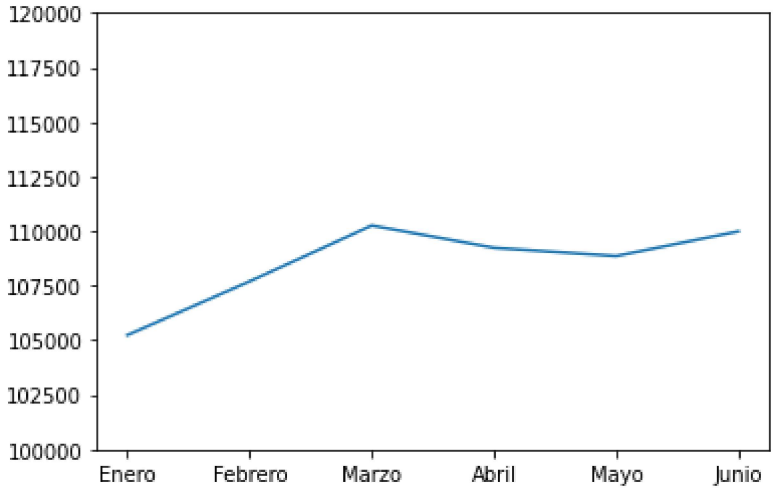
Si miramos de cerca el gráfico y los números, veremos que en marzo la facturación fue cercana a los ciento diez mil y el mes siguiente bajó a un valor cercano a los ciento nueve mil.

Mirando los números, esa baja no parece haber sido tan grande, pero cuando miramos solo el gráfico, podemos pensar que tuvimos una baja mayor.

Esto se debe a que valores de nuestro eje Y son muy próximos entre sí. Por lo tanto, cualquier cambio en los valores hacia arriba o hacia abajo puede causar distorsiones en el gráfico.

Para evitar un poco este comportamiento, podemos cambiar los valores de la base y del tope del eje Y, es decir, podemos cambiar su límite **`(ylim())`**:

```
>>> matplotlib.pyplot.plot(meses, valores)
>>> matplotlib.pyplot.ylim(100000, 120000)
>>> matplotlib.pyplot.show()
```



Al cambiar ligeramente el límite del eje Y, podemos ver que, de hecho, esas bajas en la facturación, que parecían bruscas, son más suaves de lo que parecían.

Ahora, cuando presentemos este gráfico en la reunión, será más claro para que la gente vea... ¿Vea qué? ¿De qué se trata este gráfico?

Los que lo construimos sabemos que se trata de ventas semestrales, pero ¿qué pasa con una persona que nunca ha visto este gráfico? Nuestro gráfico solo tiene números y meses.

Cuando las personas ven este gráfico, pueden preguntarse de qué se tratan estos números, ¿la cantidad de productos vendidos? ¿La ganancia en pesos o en dólares?

Para evitar estas posibles confusiones, debemos dejar claro cuál es el propósito de nuestro gráfico y qué significa cada eje.

Detallando las informaciones en el gráfico

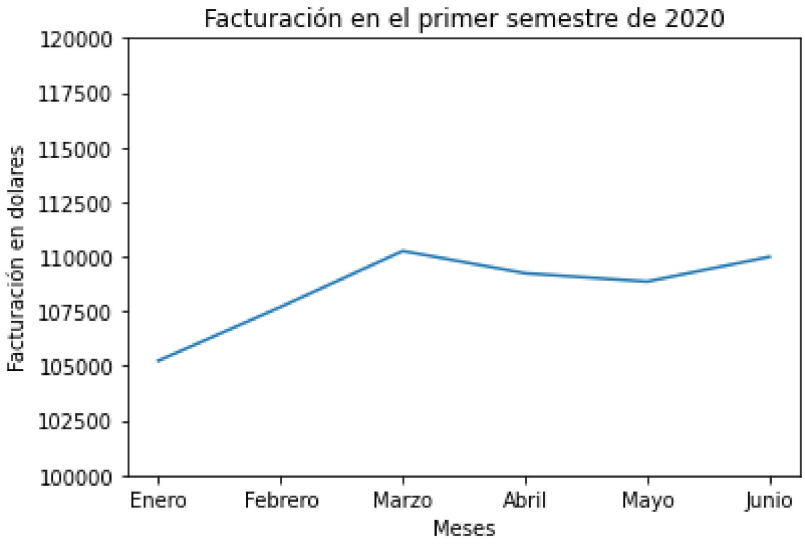
Podemos decirle al pyplot que nombre nuestro gráfico, es decir, que le agregue un título (`title`):

```
>>> matplotlib.pyplot.title('Facturación en el primer semestre de 2017')
```

También podemos pedirle al pyplot que escriba los significados de los valores en el eje **X** y en el eje **Y**. Es decir, podemos pedirle que etiquete los datos de estos ejes (**`xlabel`**, **`ylabel`**):

```
>>> matplotlib.pyplot.xlabel('Meses')
>>> matplotlib.pyplot.ylabel('Facturación en dolares')
```

Ahora, cuando le pedimos al pyplot que muestre nuestro gráfico, veremos el título y la etiqueta de cada eje:



Ahora tenemos un gráfico mucho más descriptivo. Sabemos de qué se trata el tema y conocemos el significado de los ejes **X** e **Y**.

Para saber más

Tenga en cuenta que la forma que importamos el módulo del `pyplot`, siempre que necesitamos llamar alguna función de la biblioteca, tenemos que pasar por el camino completo hasta el módulo, es decir, `matplotlib.pyplot`. Un nombre un poco grande, ¿verdad?

Por eso, una práctica habitual es dar apodos a nuestros imports. Por ejemplo, podemos decirle a Python que estamos denominando el módulo `pyplot` de `plt`, por ejemplo:

```
>>> import matplotlib.pyplot as plt
```

De esta manera, siempre que queramos llamar a este módulo, simplemente lo llamaremos por el apodo:

```
>>> plt.show()
```

Además del gráfico de líneas, pyplot también puede crear otros tipos de gráficos, como gráficos de barras, gráficos circulares, de dispersión, entre otros.

Además de crear el gráfico y agregarle título y etiqueta, también podemos cambiar el color de la línea y el color de fondo. Aumentar el tamaño de las letras, entre varias otras personalizaciones.

La parte de visualización de dados es una de las tareas más importantes de un científico de datos, porque con ella, además de entender los datos de una manera más intuitiva, nos ayuda a transmitir lo que dicen los datos a otras personas.

¿Qué tal aprender más sobre **Python** y sus diversos recursos? Entonces, ¡Mira nuestros cursos de **Python para Data Science** aquí en Alura!

ARTÍCULOS DE TECNOLOGÍA > DATA SCIENCE

BLOG

PROGRAMACIÓN
FRONT END
DATA SCIENCE
INNOVACIÓN Y GESTIÓN
DEVOPS

AOVS Sistemas de Informática S.A
CNPJ 05.555.382/0001-33

SÍGUENOS EN NUESTRAS REDES SOCIALES

ALIADOS

Empresa participante do SCALE UP ENDEAVOR

En Alura somos unas de las Scale-Ups seleccionadas por Endeavor, programa de aceleración de las empresas que más crecen en el país.

Google for Startups Alumni 2021

Fuimos unas de las 7 startups seleccionadas por Google For Startups en participar del programa Growth Academy en 2021

POWERED BY

CURSOS

Cursos de Programación

Lógica de Programación | Java

Cursos de Front End

HTML y CSS | JavaScript | React

Cursos de Data Science

Data Science | Machine Learning | Excel | Base de Datos | Data Visualization | Estadística

Cursos de DevOps

Docker | Linux

Cursos de Innovación y Gestión

Productividad y Calidad de Vida | Transformación Ágil | Marketing Analytics | Liderazgo y Gestión de Equipos | Startups y Emprendimiento